Representations and Evolutionary Operators for the Scheduling of Pump Operations in Water Distribution Networks

Manuel López-Ibáñez manuel.lopez-ibanez@ulb.ac.be
IRIDIA, CoDE,
Université Libre de Bruxelles, Brussels, Belgium

T. Devi Prasad d.p.tumula@salford.ac.uk
School of Computing, Science & Engineering,
University of Salford, UK

Ben Paechter b.paechter@napier.ac.uk
Centre for Emergent Computing, School of Computing,
Edinburgh Napier University, UK

Abstract

Reducing the energy consumption of water distribution networks has never had more significance. The greatest energy savings can be obtained by carefully scheduling the operations of pumps. Schedules can be defined either implicitly, in terms of other elements of the network such as tank levels, or explicitly by specifying the time during which each pump is on/off. The traditional representation of explicit schedules is a string of binary values with each bit representing pump on/off status during a particular time interval. In this paper, we formally define and analyze two new explicit representations based on time-controlled triggers, where the maximum number of pump switches is established beforehand and the schedule may contain less switches than the maximum. In these representations, a pump schedule is divided into a series of integers with each integer representing the number of hours for which a pump is active/inactive. This reduces the number of potential schedules compared to the binary representation, and allows the algorithm to operate on the feasible region of the search space. We propose evolutionary operators for these two new representations. The new representations and their corresponding operations are compared with the two most-used representations in pump scheduling, namely, binary representation and level-controlled triggers. A detailed statistical analysis of the results indicates which parameters have the greatest effect on the performance of evolutionary algorithms. The empirical results show that an evolutionary algorithm using the proposed representations improves over the results obtained by a recent state-of-the-art Hybrid Genetic Algorithm for pump scheduling using level-controlled triggers.

Keywords

Genetic algorithms, representation, evolutionary operators, pump scheduling, water distribution networks,

1 Introduction

Water Distribution Networks (WDNs) ensure the supply of water with adequate pressure to consumers at demand nodes. To accomplish this task, often, water must be pumped to overcome frictional losses and to store it at a higher elevation. For example, pumps lift

water from a treatment plant into an elevated tank, where water is stored temporarily before supplying it to consumers. The cost of energy consumed by pumps is a significant proportion of the total operational cost of a WDN. Pumps also need to be repaired and replaced, incurring additional costs. Carefully scheduling pump operations may reduce the total pump operational cost, which includes pumping cost and maintenance cost, while guaranteeing a competent service.

Finding optimal schedules for pumps in a WDN is a difficult task for researchers and managers alike. A careful scheduling of pump operations may shift workload to cheaper electrical tariff periods, reducing the cost of energy consumed by pumps. Furthermore, energy savings can be accomplished by pumping water when tank levels are lower, and combining the operations of several pumps efficiently. On the other hand, (future) pump maintenance costs caused by pump operations cannot be easily quantified, so surrogate measures are used to estimate it. The most common of such measures is the total number of pump switches: frequent switching (on/off) causes wear and tear of pumps and pressure surges throughout the network, and, hence, increases future maintenance costs. These maintenance costs can be considered in the optimisation problem by limiting the number of pump switches.

Most of the research in the field of water distribution optimisation has been devoted towards the optimal *design* of WDNs (Maier et al., 2003; Farmani et al., 2006; Prasad, 2009; Vasan and Simonovic, 2010). Nonetheless, there are algorithms for operational optimisation of WDNs based on linear programming (Jowitt and Germanopoulos, 1992), nonlinear programming (Chase and Ormsbee, 1993; Yu et al., 1994), dynamic programming (Lansey and Awumah, 1994; Nitivattananon et al., 1996), and heuristics (Ormsbee and Reddy, 1995; Leon et al., 2000). However, these algorithms had a limited success because of various reasons. First, the large search space combined with high evaluation time restricts the number of solutions that can be examined in a reasonable time. Second, the complexity of real-world WDNs makes the problem highly nonlinear, due to conservation of energy equations, and discontinuous, due to discrete events triggered by control rules. Therefore, the application of traditional optimisation methods, such as linear and nonlinear programming, often involve the formulation of oversimplified models of the network and the system of hydraulic equations to suit the algorithm requirements, thereby sacrificing accuracy. Algorithms based on meta-heuristics, such as evolutionary algorithms (Savic et al., 1997; van Zyl et al., 2004; Farmani et al., 2006; Rao and Salomons, 2007), simulated annealing (Goldman and Mays, 2000), and ant colony optimisation (López-Ibáñez et al., 2008), have overcome, to some degree, the above limitations, and hence, have shown promising results.

Most works on pump scheduling encode pump schedules using a binary string representing the on/off state of a pump during a pumping interval. Recently, we proposed an alternative representation based on time-controlled triggers for applying ant colony optimisation to pump scheduling (López-Ibáñez et al., 2008). In this paper, we formally define and develop this new representation and compare it with traditional representations. This new representation enables the optimisation algorithm to operate on the feasible search space, which leads to a reduction in the number of function evaluations and, hence, computation time. Moreover, the new representation provides an alternative view-point to tackle the problem. In binary representation, the algorithm tries to find the optimal status of each pump at predefined time intervals. By comparison, in the proposed time-controlled triggers method, the goal becomes finding the appropriate switch times for the pumps (in the case of *absolute* time-controlled triggers) or the appropriate duration of the operating and idle intervals (in the case of *relative* time-controlled

triggers). Therefore, we propose two variants of time-controlled triggers representation.

These two new representations require specific recombination and mutation operators in order to use them in evolutionary algorithms. We propose various operators and empirically test them using a simple evolutionary algorithm to analyse their effect. As benchmark instances, we consider two well-known instances from the literature. As far as we know, not even studies proposing specialised representations for pump scheduling have performed a comparative analysis of different representations using the same algorithm and on the same network instances. In some cases, no comparison is done at all (Wegley et al., 2000). In other cases, a new algorithm using the new representation is compared with unoptimised settings or with a previous algorithm using a different representation (Kazantzis et al., 2002). Since neither the algorithm nor the representation is the same in any step of the study, observed differences cannot be attributed exclusively to the new representation or to the algorithm. By contrast, we compare four different representations using the same algorithm. In addition, we assess the quality of the results by using as baseline for the comparison a recent state-of-the-art algorithm for pump scheduling that uses one of these four representations.

The paper is structured as follows. We first introduce the problem of scheduling pump operations in WDNs in Section 2. Next, we discuss in detail the representation of pump schedules in Section 3, where the new time-controlled triggers representation is proposed. Section 4 proposes new recombination and mutation operators for the two variants of time-controlled triggers, and describes the operators used in the experiments for the other representations. Section 5 presents a statistical analysis of different parameter settings for each representation in order to identify adequate settings for a fair comparison of the different representations, which is given in Section 6. Finally, we present our conclusions and suggest potential research directions in Section 7.

2 Pump Scheduling Problem

The main goal in the pump scheduling problem is to minimise the cost of pumping water, while satisfying physical and operational constraints (Ormsbee and Lansey, 1994), by means of scheduling pump operations over a *scheduling period*, typically 24 hours. There are two types of costs associated with the operation of pumps: energy costs and maintenance costs.

The energy cost may be composed of an *energy consumption charge* (\$/kWh), i.e., the cost of electric energy consumed during a time interval, and a *demand charge* (\$/kW), i.e., the cost associated with the maximum amount of power consumed within a billing period (Walski et al., 2003). Formally, let the operations of N^{P} pumps be scheduled over a time period. This *scheduling period* is divided into a number of time intervals (N^{T}). A certain pump schedule s describes (explicitly or implicitly) which pumps operate during which time interval. The total cost of energy is calculated as:

$$C_{\mathrm{E}}(s) = \sum_{n=1}^{N^{\mathrm{P}}} \left[P_{\mathrm{d}}\, E_{\mathrm{d}}(n) + \sum_{t=0}^{N^{\mathrm{T}}} P_{\mathrm{c}}(t)\, E_{\mathrm{c}}(n,t)\, s(n,t) \right] \tag{1}$$

where $s(n,t)$ = duration for which pump n is operating during time interval t (hour)
$P_{\mathrm{c}}(t)$ = energy consumption tariff during time interval t (\$/kWh)
$E_{\mathrm{c}}(n,t)$ = energy consumption rate of pump n during time interval t (kWh/h)
P_{d} = demand charge (\$/kW)
$E_{\mathrm{d}}(n)$ = maximum electric power consumption of pump n (kW)

The energy consumption rate of a pump depends on the flow through the pump, the head supplied by the pump, which is measured as the energy or pressure of a vertical column of water of a certain height in meters, and the efficiency at which it operates, during a particular time interval (Walski et al., 2003):

$$E_{\mathrm{c}}(n,t) = \frac{10^{-3} \cdot \gamma \cdot Q(n,t) \cdot h(n,t)}{e(n,t)} \qquad \text{(kWh/h)} \tag{2}$$

where γ = specific weight of water (N/m^3)
$Q(n,t)$ = flow rate through pump n during time interval t (m^3/s)
$h(n,t)$ = total dynamic head supplied by pump n during time interval t (m)
$e(n,t)$ = wire-to-water efficiency of pump n during time interval t (%)

The maximum electrical power consumed by pump n during the scheduling period is calculated as:

$$E_{\mathrm{d}}(n) = \frac{10^{-3} \cdot \gamma \cdot Q^{\max}(n) \cdot h(n)}{e(n)} \qquad \text{(kW)} \tag{3}$$

where γ = specific weight of water (N/m^3)
$Q^{\max}(n)$ = peak flow rate through pump n (m^3/s)
$h(n)$ = total dynamic head supplied by pump n (m)
$e(n)$ = wire-to-water efficiency of pump n (%)

The demand charge is normally applied to the maximum power demand (kW) over a billing period (e.g. a month) longer than the scheduling period (e.g. a day). Nonetheless, maximum demand policies can be modelled within a scheduling period by calculating the corresponding penalty cost to the maximum power used over the scheduling period (McCormick and Powell, 2003a).

Maintenance costs, on the other hand, are difficult to quantify. Instead, they are estimated using a surrogate measure, such as the utilisation duration of pumps or number of pump switches. Reducing the utilisation of the pumps is mainly achieved by minimising the energy cost. On the other hand, there is no such direct relationship between pump switches and energy cost. A *pump switch* is defined as the action of turning on a pump that was not operating during the previous time interval (Lansey and Awumah, 1994). Frequent switching causes wear and tear of pumps, which, in turn, increases maintenance costs. Moreover, pump switches generate pressure surges that cause an unspecified damage to network components, such as pipes and valves. Thus, the general practice is to minimise the number of pump switches in order to reduce both the wear of the pumps and the damage to pipes and valves, hence, minimising future maintenance costs. Most works consider energy cost as the most important objective, and add the number of pump switches as a constraint to the problem (Lansey and Awumah, 1994; Mäckle et al., 1995; van Zyl et al., 2004; López-Ibáñez et al., 2008).

2.1 Constraints of the Pump Scheduling Problem

In order to be practical, feasible schedules must satisfy certain constraints. These constraints include *hydraulic constraints*, also called system constraints, which define the hydraulic equilibrium state of the system, e.g., Conservation of Mass at each node and Conservation of Energy around each loop in the network. On the other hand, *bound constraints* represent system performance criteria. They include constraints on junction pressures, pipe flow rates or velocities, and tank water levels.

Constraints on tank water levels typically include minimum and maximum limits on tank levels, and balance between supply and demand from tanks. Minimum and maximum tank limits may be explicit constraints of the problem or can be implicitly enforced by a hydraulic simulator. Balance between water supplied to and consumed from tanks is achieved by ensuring that tanks recover their levels by the end of scheduling period. Balancing supply and demand also allows operators to apply a similar pump schedule to the next scheduling period, assuming consumer demands and network conditions are similar in consecutive periods.

For a perfect periodicity of the operations, the volume of water in each tank by the end of the scheduling period must be equal to its initial volume (Cohen, 1982). However, this is a very strict constraint. A relaxed formulation allows the final volume of water at each tank to be different from its initial volume and simply requires that the total volume of water pumped into the network is the same as the amount consumed (Goldman and Mays, 2000). This condition ensures the balance between water supply and demand. However, water may be stored at a lower elevation by the end of the scheduling period than at the start. Such a difference in elevation implies a loss of energy in the system and prevents periodicity. An alternative formulation that does not allow such energy losses requires that the final volume in a tank should not be lower than its initial volume (Mäckle et al., 1995; van Zyl et al., 2004).

Following this latter definition, we formulate the constraint on the balance between supply and demand as follows. *Tank volume deficit* (ΔV_k) is defined as the difference in percentage between the initial volume ($V_{k,S}$) and the final volume ($V_{k,E}$) of water in a tank k (4a). A negative volume deficit represents a surplus of water in the tank. However, we do not assume that this surplus compensates the loss of water in a different tank. The *volume deficit tolerance* (ΔV^{tol}) is a parameter of the problem formulation that defines the volume deficit that is allowed (4b). Only values that are higher than ΔV^{tol} are accumulated to calculate the *total volume deficit* (ΔV) of a particular schedule, which must be zero in a feasible solution (4c).

$$\Delta V_k = 100 \cdot \frac{V_{k,S} - V_{k,E}}{V_{k,S}} \tag{4a}$$

$$\Delta V'_k = \begin{cases} \Delta V_k & \text{if } \Delta V_k > \Delta V^{\text{tol}}, \\ 0 & \text{otherwise.} \end{cases} \tag{4b}$$

$$\Delta V = \sum_{k=1}^{N^{\text{t}}} \Delta V'_k \;=\; 0 \tag{4c}$$

where N^{t} is the number of tanks in the network. The parameter ΔV^{tol} allows operators to model different scenarios (López-Ibáñez, 2009). However, the most common setting is $\Delta V^{\text{tol}} = 0$, that is, no volume deficit is allowed, which is the setting used by the instances considered in this paper.

In addition to balancing supply and demand, a reliable network service must supply water to consumers at adequate pressures. Therefore, the optimisation model must satisfy minimum pressure constraints at demand nodes:

$$H_{k,t} \geq H_k^{\min} \tag{5}$$

where $H_{k,t}$ is the head supplied at node k during time period t, measured as the pressure of a vertical column of water of a certain height in meters, and $H_k^{\min}$ is the minimum head required at node k. In particular, we accumulate the violations of the above constraints into a single pressure deficit constraint:

$$\Delta H_{k,t} = \begin{cases} \frac{H_k^{\min} - H_{k,t}}{H_k^{\min}} & \text{if } H_{k,t} < H_k^{\min}, \\ 0 & \text{otherwise.} \end{cases}$$
$$\Delta H = \sum_{t=1}^{N^{\mathrm{T}}} \sum_{k=1}^{N^{\mathrm{d}}} \Delta H_{k,t} \quad = 0 \tag{6}$$

where N^{T} is the number of time periods and N^{d} is the number of demand nodes.

Finally, an additional constraint on the number of pump switches of a schedule is often incorporated to the problem formulation in order to reduce maintenance costs, as mentioned above. However, our goal is not limiting the total number of pump switches (N^{sw}), but actually limiting the number of switches of each pump (N_p^{sw}). The difference is that by limiting N^{sw}, a schedule may still contain a pump with an excessive number of switches, whereas this cannot occur if the constraint is applied to N_p^{sw} as follows:

$$N_p^{\mathrm{sw}} \leq SW \qquad \forall p \in \{1, \ldots, N^{\mathrm{p}}\} \qquad \text{where } N^{\mathrm{sw}} = \sum_{p=1}^{N^{\mathrm{p}}} N_p^{\mathrm{sw}} \tag{7}$$

where SW is a constant to be specified that denotes the maximum number of switches allowed per pump during the scheduling period.

The above constraints are the most frequently used constraints. Additional constraints, such as limits on source flows or velocity constraints, may be incorporated to the problem formulation depending on particular requisites of a network and the optimisation approach. In particular, EPANET (Rossman, 1999), the hydraulic simulator that we use later for evaluating candidate pump schedules,[1] may issue *warnings* for specific undesirable situations (Rossman, 2000). Such warnings indicate that the schedule is problematic and should not be considered a feasible solution to the problem. Therefore, we add an additional constraint that requires feasible solutions to generate no simulation warnings.

2.2 Constraint handling methodology

Previous works on pump scheduling have dealt with constraints by penalising the objective function (Mäckle et al., 1995; Goldman and Mays, 2000; van Zyl et al., 2004). The penalty function method imposes a fixed trade-off between the amount of constraint violation and the value of the objective function. Low penalty values would allow large constraint violations in return for small reductions in the objective value, while higher penalty values would require a larger decrease of the objective value to compensate the same amount of constraint violation. Moreover, different penalty values are required for different types of constraints and the degree of violation of some of these constraints cannot be easily quantified. Penalty values, in general, are obtained either using ad-hoc techniques or by trial-and-error, requiring additional fine-tuning and experimental

[1] We use our own GNU/Linux port of EPANET that maintains backwards compatibility. It is available for download at `http://iridia.ulb.ac.be/~manuel/epanetlinux`

runs of the particular algorithm. Furthermore, penalty values that are optimal for one network instance are unlikely to be appropriate for a different network.

For these reasons, we avoid the use of a penalty function and adopt a simpler and more general constraint handling method based on ranking solutions with respect to their constraint violations (Deb, 2000). In this ranking method, given two candidate solutions, the criteria to choose the best solution are:

1. select the solution with the lowest pressure violation (ΔH in Eq. 6);

2. if pressure violations are equal, select the solution with the lower number of warnings from the simulator;

3. for equal number of warnings, select the solution with the lower total volume deficit (ΔV in Eq. 4);

4. if total volume deficits are equal, select the solution with the lowest objective function value, that is, lowest electricity cost (C_{E} in Eq. 1).

In the case of the binary and level-controlled triggers representations, the constraint on the number of pump switches ($N_p^{\mathrm{sw}} \leq 3$, Eq. 7 with $SW = 3$) is explicitly incorporated to our constraint handling method. In particular, out of two solutions that violate this constraint, the one containing the pump with the highest number of switches is considered worse. This constraint is given a lower priority than the constraint on volume deficit, and thus it is considered after all the other constraints.

These criteria effectively rank a feasible solution (zero total volume deficit, no warnings and no pressure violations) better than any infeasible one. Feasible solutions are compared with respect to their objective function values only, and infeasible solutions are compared according to their degree of infeasibility. The order chosen for the comparison of constraint violations establishes some preferences. A solution with volume deficit, where enough water is supplied to meet the demand but a balance is not achieved by the end of the simulation, is preferred over a solution having pressure violations, where the adequate demand cannot be supplied. Warnings from the simulator are considered to be worse than volume deficit, since warnings indicate some problem preventing the correct evaluation of the solution by the simulator (e.g., a pump was forced to shut down because it could not deliver enough head). However, a solution that generates warnings and no pressure violations is preferred over a solution that has pressure violations. We observed that a small modification to such a solution often removes the warnings, without significantly altering the electrical cost. On the other hand, a solution with pressure violations would require larger changes, since pumps need to be active for more hours to supply the demand at required pressures.

3 Representation of pump schedules

A solution to the pump scheduling problem is any possible schedule of pumps for a predefined scheduling period, typically 24 hours. A particular representation describes how a sequence of decision variables maps to a pump schedule. For the sake of clarity, we assume that the *sequence* of decision variables representing a schedule of the pumps has the following general form $S = \{s^1, s^2, \ldots, s^{N^{\mathrm{P}}}\}$ where each s^p corresponds to the *sequence* of decision variables representing the schedule of pump p. This formulation has the advantage that we can assume a single pump when discussing different representations, while extending the discussion to several pumps is straightforward.

A schedule can be represented either explicitly or implicitly. *Explicit* representations define schedules by directly specifying the status of each pump (Mäckle et al., 1995; Pezeshk and Helweg, 1996; Wegley et al., 2000; Sakarya and Mays, 2000; Goldman and Mays, 2000; Savic et al., 1997; McCormick and Powell, 2003b, 2004). Typically, *explicit* representations divide the operation period into several time intervals. For each time interval, a decision variable either indicates the status of the pump (active, idle or speed for variable frequency drive pumps) or the fraction of time a pump is operating during that time interval. On the other hand, *implicit* representations define the operations of pumps in terms of properties of other elements of the network (Dandy and Gibbs, 2003; van Zyl et al., 2004; Kazantzis et al., 2002; Atkinson et al., 2000). For example, water levels in tanks are often used to trigger operation of pumps. Hence, the goal becomes one of determining optimal values of those surrogate variables.

In the following section, we first discuss two well-known and widely used representations: the binary representation and the representation based on level-controlled triggers. Then, we formally introduce two representations based on the concept of time-controlled triggers. Although other representations have been suggested in the literature, none of them has been shown to give a clear advantage over both the binary representation and level-controlled triggers in the general pump scheduling problem. In fact, they were typically developed to deal with specific requirements or limitations of the formulation of the problem, the optimisation algorithm or the simulation model.

3.1 Binary Representation

The binary representation is the most commonly used explicit representation of pump schedules (Mäckle et al., 1995; Savic et al., 1997; Goldman and Mays, 2000; Sotelo et al., 2002). In the binary representation, the scheduling period is divided into a fixed number (N^{T}) of smaller intervals. A single binary value is used to represent the status of a pump during each interval, and equals to one if the pump is active during the time interval, or zero if the pump is idle.

$s^p =$

t_0	t_1	t_2	t_3	t_j	$t_{N^{\mathrm{T}}}$
0/1	0/1	0/1	0/1	$\cdots$	0/1

Figure 1: Binary representation of a pump schedule.

Given a particular solution, the number of pump switches is the number of 01 sequences plus one if the schedule starts with 1 and ends with 0. Thus, the maximum number of switches per pump is $N_p^{\mathrm{sw}} \leq \lfloor N^{\mathrm{T}}/2 \rfloor$. The size of the search space depends on both the number of time intervals and the number of pumps, being the total number of possible solutions $2^{(N^{\mathrm{T}} \cdot N^{\mathrm{P}})}$.

The binary representation can only represent schedules where pump switching occurs at multiples of ratio T/N^{T}. For example, in the typical 24 one-hour intervals, the status of a pump cannot change in the middle of a one-hour period, thus an operating interval may last two or three hours but not 2.5 hours. This limits the flexibility of the schedules. The flexibility can be increased just by using a larger N^{T}, but this exponentially enlarges the search space.

3.2 Level-controlled Triggers

The operation of a pump can be triggered at certain water levels of a storage tank. Typically, a pair of trigger levels, lower and upper, are set up in a tank such that when water level falls below or goes above the respective level, the pump activates or stops.

Level-controlled triggers are used to keep the water level in a tank within an operating range that is a reduced interval of a contractual range. If the water level falls below or goes over the levels of this contractual range, then a penalty cost may exist. This penalty cost does not exist in our application, since the EPANET hydraulic simulator prevents the water level from falling outside the contractual range by closing the tank riser. However, this may result in violations of pressure constraints, because pressure at some of the demand nodes decreases when a tank cannot supply water.

Electricity tariff is typically divided into peak and off-peak periods, and different pairs of level-controlled triggers are used for each period. Thus, a solution in level-controlled triggers representation has the following formulation:

$$s^p = \begin{array}{cccc} & \text{peak} & & \text{off-peak} \\ \hline H_{\text{lo}} & H_{\text{up}} & H'_{\text{lo}} & H'_{\text{up}} \\ \hline \text{lower} & \text{upper} & \text{lower} & \text{upper} \end{array}$$

Figure 2: Level-controlled triggers.

Each trigger level is constrained to values within the contractual range of the tank. Additionally, for each pair of lower/upper trigger levels, the lower level is never higher than the corresponding upper level ($H_{\text{lo}} \leq H_{\text{up}}$ and $H'_{\text{lo}} \leq H'_{\text{up}}$).

There is no limit in the number of pump switches that the level-controlled triggers representation may generate. Apart from the fact that narrow operating ranges will, in general, result in more pump switches than wider ranges, there is no rule of thumb that can be applied to estimate the number of pump switches generated by a particular setting of level-controlled triggers. In fact, excessive number of pump switches is often a problem in applications using level-controlled triggers. This is sometimes acknowledged as a potential problem, but no strategies are considered to prevent it (Kazantzis et al., 2002). In other cases, the schedules obtained by the optimisation algorithm are manually fine-tuned a posteriori to reduce the number of pump switches (Atkinson et al., 2000). A better approach incorporates an explicit constraint on the number of pump switches into the optimisation algorithm and constraint handling techniques are applied, e.g., penalising the objective function (electrical cost) proportionally to the number of pump switches (van Zyl et al., 2004).

3.3 A New Representation for Pump Scheduling: Time-controlled Triggers

We recently used the concept of time-controlled triggers to develop an explicit representation in order to apply ant colony optimisation to pump scheduling (López-Ibáñez et al., 2008), with the limitation that only schedules with an exact pre-defined number of switches were representable. In this section, we formalise the definition of two variants of the *time-controlled triggers* representation that do not have such artificial limitations. These two representations have the advantage of implicitly satisfying the constraint on the number of pump switches, while allowing less switches than the limit.

In contrast to the binary representation, which encodes the status of a pump during each time interval, the *time-controlled triggers* representation encodes the time when a pump changes its status. The concept of encoding time has already been proposed in the literature. Sakarya and Mays (2000) and McCormick and Powell (2004) used continuous variables to encode the proportion of time that a pump (or combination of pumps) is active during a time interval. However, our proposal allows us to directly encode pump switches in the representation. A pump switch is defined as turning on a pump that

was previously off (Lansey and Awumah, 1994). However, in a *periodic* schedule each pump switch actually implies two changes in the status of a pump: *off* to *on* for the pump switch and *on* to *off* to achieve the situation previous to the pump switch. For example, a schedule where a pump is initially on and is never turned off during the whole scheduling period does not contain any pump switch or status change. On the other hand, a schedule where a pump is initially on and it is eventually turned off contains one pump switch and two status changes.

Therefore, a pair of decision variables is required to define a pump switch in time-controlled triggers representation. We can thus limit the maximum number of switches per pump simply by limiting the number of decision variables of each solution. For example, a schedule of a single pump in a *time-controlled triggers* representation of length six will allow a maximum of three pump switches. In general, for a maximum of SW switches per pump, there will be $2 \cdot SW$ decision variables for each pump.

$$s^p = \boxed{t_1 \mid t'_1 \parallel t_2 \mid t'_2 \parallel \ldots \mid \ldots \parallel t_{SW} \mid t'_{SW}}$$

Figure 3: Time-controlled triggers for pump p with SW pump switches.

The range of the decision variables depends on the precision with which we want to measure time and on the time horizon. If continuous values are used, then the precision is arbitrary. On the other hand, discrete values may be preferred in order to limit search space and because arbitrary time precision is impractical. Therefore, assuming a time horizon of 24 hours, the range of decision variables may be an integer within $[0, 86400]$ for a precision of seconds, $[0, 1440]$ for a precision of minutes, and $[0, 24]$ for a precision of hours. For simplicity and following the typical binary representation, we will assume intervals of one hour.

This new representation enables the optimisation algorithm to conduct the search in the feasible region of the search space. For example, let us consider the schedule of a single pump for 24 intervals of one hour. In the binary representation, each interval can have either of the two states (on/off) and, thus, there are 12 possible pump switches. The search space contains $2^{24} = 16\,777\,216$ candidate solutions. However, if the number of pump switches is restricted to three ($N_p^{\text{sw}} \leq 3$), the feasible search space with respect to this constraint is 290 998 solutions, which is less than 1.74% of the total search space. Table 1 gives the number of potential solutions with respect to the number of pump switches. These values were obtained by explicitly enumerating all 2^{24} possible schedules of a single pump.

The size of the search space grows with N^{T}. For example, when considering $N^{\text{T}} = 48$, that is, 30 minutes intervals in a daily scheduling period, there are 24 934 442 possible schedules of a single pump with three or less pump switches. This is more than 85 times the number of solutions for $N^{\text{T}} = 24$. On the other hand, it is only 8.86×10^{-6} percent of the total number of possible schedules for a single pump with $N^{\text{T}} = 48$. That is, on the one hand, the size of the feasible search space grows with N^{T}, but on the other hand, it grows far slower than the total search space. With additional pumps, the search space grows exponentially. For just two pumps there are $2^{24} \cdot 2^{24} = 2^{48}$ potential schedules in total, of which only 84 679 836 004 potential solutions (0.03%) satisfy the constraint $N_p^{\text{sw}} \leq 3$. Although the number of potential solutions increases dramatically for higher number of pumps, the feasible search space becomes a smaller fraction of the total search space.

The time-controlled triggers representation may be implemented in at least two different ways, depending on whether the time encoded by the representation is *absolute*

Table 1: Search space size for the scheduling of a single pump in 24 hours with respect to various limits on the number of pump switches (SW).

SW	$N_p^{\text{sw}} = SW$		$N_p^{\text{sw}} \leq SW$	
	Feasible space	% of total	Feasible space	% of total
1	552	0.0033	554	0.0033
2	21252	0.1267	21806	0.13
3	269192	1.6045	290998	1.7345
4	1470942	8.7675	1761940	10.502
5	3922512	23.38	5684452	33.882
6	5408312	32.2361	11092764	66.118
12	2	0.00001	16777216	100

time since the start of the scheduling period, or *relative* to a previous change of pump status. Each of these implementations may lead to different results. Moreover, we do not want to impose an exact number of pump switches, like in earlier formulations of time-controlled triggers (López-Ibáñez et al., 2008), but a maximum limit. Therefore, we propose a formulation which is able to represent schedules with fewer switches than the number of pairs of decision variables. In the following sections, we will examine how to achieve this for two variants of time-controlled triggers.

3.3.1 Absolute Time-controlled Triggers

When decision variables are absolute time, each decision variable represents the time from the start of scheduling period at which the status of a pump changes. Let us assume for now that pumps are off at the start of the scheduling period. We consider that t_i corresponds to a transition from *off* to *on*, and t'_i corresponds to a transition from *on* to *off*. Therefore, a pair of decision variables $\langle t_i, t'_i \rangle$ represents an operating interval during which the pump is active. Figure 4 shows an example schedule represented with absolute time-controlled triggers, where each value is a number of hours since the start of the scheduling period (7 *am*).

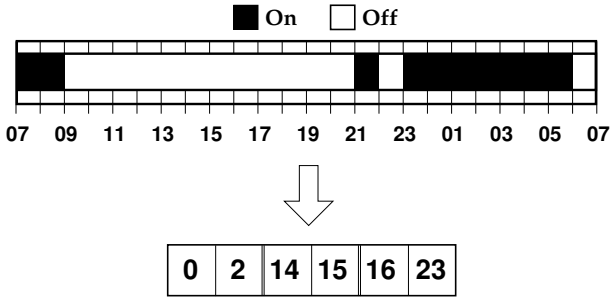

Figure 4: Example of absolute time-controlled triggers representation.

A possible formulation of the above description would be:

$$s^p = \{\langle t_1, t'_1 \rangle, \langle t_2, t'_2 \rangle, \ldots, \langle t_{SW}, t'_{SW} \rangle\} \tag{8}$$
$$\forall i \in \{1, \ldots, SW\} \quad t_i, t'_i \in [0, T]$$
$$\forall i \in \{1, \ldots, SW - 1\} \quad t_i < t'_i < t_{i+1} < t'_{i+1}$$

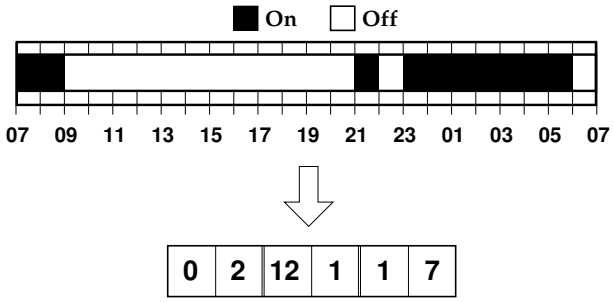

Figure 5: Example of relative time-controlled triggers representation.

The inclusion of the value 0 in the range of values makes unnecessary the previous assumption that pumps are off at the start of the scheduling period, since a pair $\langle 0, t'_1 \rangle$ represents an operating interval where the pump is active at the start of the scheduling period. However, this formulation is too strict in the sense that it cannot represent schedules with less than SW pump switches ($N_p^{\text{sw}} \leq SW$). In order to represent such schedules we need to introduce *empty operating intervals*, that is, an operating interval that has no effect whatsoever on the schedule apart from reducing the number of switches in a solution. We denote empty intervals with the special symbol $\langle -, - \rangle$. Thus, we extend the formulation above in the following way:

$$s^p = \{\langle t_1, t'_1 \rangle, \langle t_2, t'_2 \rangle, \ldots, \langle t_{SW}, t'_{SW} \rangle\} \tag{9}$$

$$\forall i \in \{1, \ldots, SW\} \quad \langle t_i, t'_i \rangle \in \langle [0, T], [0, T] \rangle \cup \langle -, - \rangle$$
$$\forall i \in \{1, \ldots, SW - 1\} \quad t_i < t'_i < t_{i+1} < t'_{i+1} \quad \text{iff } \langle t_i, t'_i \rangle \neq \langle -, - \rangle$$

3.3.2 Relative Time-controlled Triggers

If decision variables are relative time intervals, each pair of decision variables represents the time during which a pump is inactive and active, respectively. According to this, the sequence of decision variables $\langle t_i, t'_i \rangle$ implies a change of state from an inactive one (during t_i) to an active one (during t'_i), and, hence, it also implies a single switch.

It immediately follows from this definition that the sum of all time intervals for each pump must be less than or equal to the scheduling period T. By allowing the sum to be less than T, we can represent schedules where the pump is not active by the end of the scheduling period. If we let the range of valid values for any decision variable be $[0, T]$, then zero-length time intervals are allowed, thus enabling the representation of schedules with less than or equal to SW pump switches ($N_p^{\text{sw}} \leq SW$):

$$s^p = \{\langle t_1, t'_1 \rangle, \langle t_2, t'_2 \rangle, \ldots, \langle t_{SW}, t'_{SW} \rangle\} \tag{10}$$

$$\forall i \in \{1, \ldots, SW\} \quad t_i, t'_i \in [0, T] \quad \text{and} \quad \sum_{i=1}^{SW} (t_i + t'_i) \leq T$$

Figure 5 shows an example of relative time-controlled triggers representation. Each value is the number of hours that the pump is either inactive or active.

4 Evolutionary Operators

In this paper, our goal is to study the effect of several representations and their corresponding evolutionary operators, hopefully minimising the influence of other algorithmic

factors. Therefore, we make use of a very basic evolutionary algorithm, henceforth called the simple evolutionary algorithm (SEA). The algorithm starts with the initialisation of the main population P_{all} with α random solutions. Then, μ solutions from the main population are selected as parents using a binary tournament. A recombination operator is applied to pairs of parents in order to generate μ offspring solutions. Mutation is applied to each decision variable of each offspring with a certain probability. The μ new solutions generated are evaluated to calculate the objective function value and constraints. These new solutions replace the μ worst solutions in the main population. As long as $\alpha > \mu$, the best solution found will always be present in the population, implementing *elitism*. The larger the difference between α and μ, the stronger the elitism. Evolutionary algorithms applied to pump scheduling often use very strong elitism (Savic et al., 1997; van Zyl et al., 2004), and we follow this setting here.

There are many evolutionary operators described in the Evolutionary Computation literature. Michalewicz (1996) and Herrera et al. (2003) provide extensive surveys. Since operators work directly on the representation of solutions, there are some differences between the operators used for each particular representation. We first briefly describe the operators chosen for the binary and level-controlled triggers representations. Then, we introduce custom operators for the new time-controlled triggers representation.

4.1 Binary Representation

For the binary representation, we focus on three well-known types of recombination: *one-point*, *two-point* and *uniform* crossover. In the case of one- and two-point recombination, the schedules are recombined per pump by using the same crossover point for each pump. That is, given a crossover point $k \in [1, N^{\mathrm{T}} - 1]$, the offspring schedule is formed by combining, for each pump p, the schedule of pump p from one parent up to time interval k and the schedule of the same pump from the other parent from time interval $k + 1$ up to N^{T}. This approach does not seem to have been explicitly used in previous algorithms using the binary representation, which simply divided the whole binary string containing the schedule of all pumps (Mäckle et al., 1995; Savic et al., 1997). This latter approach disregards the interactions between the pumps and the fact that all pumps have an effect at the same time over the network. By comparison, our approach defines building blocks in terms of the status of all pumps during a time interval, because the result of a simulation step is influenced by the combined status of all pumps at once. In a sense, the schedules of the pumps are applied "in parallel" to the network. Our intuition is that the proposed crossover captures the "parallel" nature of the problem better. Strictly speaking, our approach could be called one-point-per-pump and two-point-per-pump. However, for brevity we will refer to them simply as one-point and two-point crossover.

Mutation is performed using *flip* mutation operator, which reverses the status of a pump at a particular time interval. For each offspring solution, the mutation operator is applied to each time interval of each pump with a certain mutation probability. In the binary representation, there are $N^{\mathrm{p}} \cdot N^{\mathrm{T}}$ decision variables per solution, and hence, a probability of mutation of $1/(N^{\mathrm{p}} \cdot N^{\mathrm{T}})$ would mutate one time interval per solution. In earlier experiments, we found out that this mutation rate was too low, and better results were obtained with a mutation rate of two intervals modified per solution, that is, with a mutation probability equal to $2/(N^{\mathrm{p}} \cdot N^{\mathrm{T}})$.

4.2 Level-controlled Triggers

In the case of level-controlled triggers representation, the variables are real numbers representing water level of a particular tank. Therefore, we use three well-known recom-

bination operators for real-valued variables:

- *Rand-arithmetical*, where each decision variable i in the offspring schedule c is calculated from the parent schedules a and b as $c_i = \lambda a_i + (1 - \lambda)b_i$, where λ is a random value between 0 and 1. It is also known as *line recombination* (Mühlenbein and Schlierkamp-Voosen, 1993) or *arithmetical recombination* (Michalewicz, 1996, p. 128).
- *Average recombination* is equivalent to arithmetical recombination with $\lambda = 0.5$, that is, $c_i = (a_i + b_i)/2$.
- *Extended intermediate recombination* is also known as BLX-α with $\alpha = 0.25$ (Eshelman and Schaffer, 1992). Each offspring level c_i is a value randomly chosen from the interval $[c_{\min} - I \cdot 0.25, c_{\max} + I \cdot 0.25]$, where $c_{\max} = \max\{a_i, b_i\}$, $c_{\min} = \min\{a_i, b_i\}$ and $I = c_{\max} - c_{\min}$.

We test three mutation operators:

- *Uniform mutation*, or *random mutation* (Michalewicz, 1996), replaces a value by a new random value from the allowed domain. In our case, if the value is an upper trigger, it is replaced by a random value between the lower trigger and the maximum level of the corresponding tank. For a lower trigger, the interval is from the minimum level to the upper trigger.
- *Replace mutation* is a less restricted version of *uniform mutation*, where a trigger can take any value within the limits of the corresponding tank. The procedure that ensures that the mutated solution is valid is described below.
- *Gaussian mutation* (Fogel, 1995) modifies a value by adding some amount of gaussian noise. For the normal (Gaussian) distribution about 99.7% of values are within three standard deviations. Therefore, if we replace a trigger level c_i with a new value obtained from the normal distribution $\mathcal{N}(c_i, \sigma)$, the new value would be within the maximum and minimum water levels of the tank k ($H_k^{\text{range}} = H_k^{\max} - H_k^{\min}$) associated with the level-controlled trigger i if $H_k^{\min} + 3\sigma \leq c_i \leq H_k^{\max} - 3\sigma$. As a rule of thumb, we want to guarantee that the new value is within the limits of the tank whenever the original value was within the range $[H_k^{\min} + \frac{1}{4}H_k^{\text{range}}, H_k^{\max} - \frac{1}{4}H_k^{\text{range}}]$. Hence, we choose $\sigma = \frac{1}{12}H_k^{\text{range}}$. A larger σ would generate more often values outside the valid range, whereas a smaller σ would generate values closer to c_i.

The above mutation operators are applied with a probability equal to one divided by the length of the solution, i.e., $1/(4 \cdot N^{\text{P}})$, since for each pump there are four trigger levels. Hence, the expected number of trigger levels modified per solution is one.

After recombination and mutation, we ensure that the values do not exceed any tank's operational levels by setting values over the maximum to the maximum level and values below the minimum to the minimum level. Moreover, if the value of the lower trigger is higher than the value of the corresponding upper trigger, the values are exchanged.

4.3 Absolute Time-controlled Triggers

Custom recombination and mutation operators are required for the time-controlled triggers representation in order to maintain the implicit constraint on the number of pump switches and other representation constraints (see Section 3.3.1). We developed variants

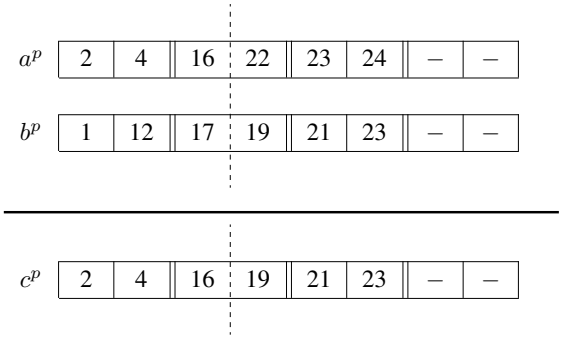

Figure 6: Example of one-point recombination for absolute time-controlled triggers.

of *one-point*, *two-point*, *uniform* and *rand-arithmetical* recombination, and of *uniform* and *replace* mutation. These operators are applied for each pump's schedule in a solution. For *absolute time-controlled triggers*, the recombination operators are adapted as follows:

- *One-point recombination.* Given a crossover point $k \in [1, 2 \cdot SW - 1]$, the schedule of each pump p in the offspring solution is obtained by joining the values from 1 to k of the first parent and the values from $k + 1$ to $2 \cdot SW$ of the second parent. For example, schedule of pump p in offspring c is obtained from two parents a and b with ($SW = 4$, $k = 3$) as shown in Fig. 6.

  This recombination may eventually generate solutions with invalid representations, since it may break the increasing order of the values. To keep the order, the values are sorted after recombination. This may result in two equal successive values, which are replaced by empty switches $\langle -, - \rangle$, and moved to the end of the schedule. This one-point recombination can be straightforwardly extended to n-point recombination. We experimentally study both *one-point* and *two-point* variants.

- *Uniform recombination.* The idea of uniform crossover is modified to suit the absolute time-controlled triggers representation in the following way. First, combine the triggers for both parents and keep track of the status (on/off) of each parent at each time interval. Next, randomly select one status, following the traditional uniform recombination. Finally, merge contiguous time intervals with the same status into larger time intervals by removing the intermediate trigger values. The resulting trigger values are used to construct the offspring solution. The following example illustrates the procedure.

  Let us assume the schedules of pump p in two parents a and b shown in Fig 7a. In absolute time-controlled triggers, each trigger value represents the time of the day at which a pump is turned on/off. Pumps initial status is assumed to be inactive (a pump initially active has 0 as its first trigger value). First, the trigger values of both parents are combined into one single time line (Fig 7b). The horizontal axis of this time line contains as many time intervals as there are different trigger values in the parents. Trigger values that do not appear in either parent are not taken into account. The time line describes the status (on/off) during each time interval of each parent given in the vertical axis. In the next step, for each time interval, one single status is randomly selected with equal probability among the status of the parents. Let us assume that the values marked in bold were the ones randomly selected,

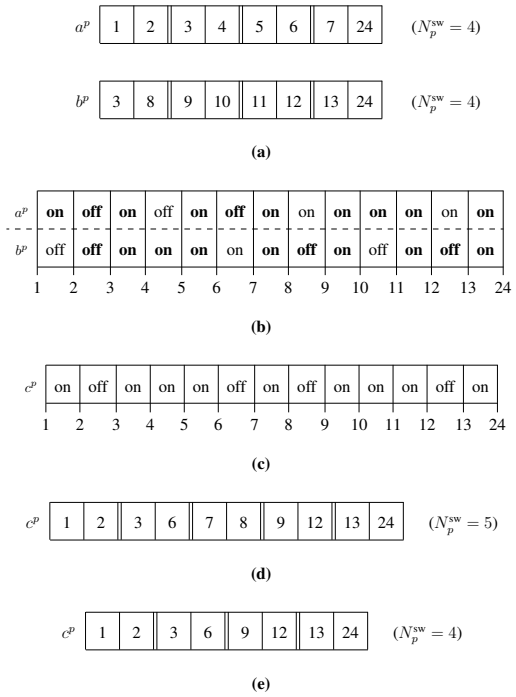

Figure 7: Example of uniform recombination for absolute time-controlled triggers representation.

producing the combined time line shown in Fig 7c. Finally, contiguous intervals with the same on/off state are merged into larger time intervals, and the boundaries of the intervals are used to construct the time-controlled triggers representation of the offspring solution (Fig 7d). The resulting schedule may have any number of pump switches, hence breaking the implicit constraint on pump switches. This is what occurs in Fig 7d. In this case, the trigger values corresponding to the shortest time interval are successively eliminated from the solution until it contains SW pump switches (Fig 7e).

- *Rand-arithmetical recombination.* This is similar to the rand-arithmetical recombination described for level-controlled triggers. For each pump p, the offspring schedule c^p is calculated from the parent schedules a^p and b^p as $c_i^p = \lambda a_i^p + (1-\lambda) b_i^p$, where λ is a random value between 0 and 1 and $i \in \{1, \ldots, 2 \cdot SW\}$. As a special case, if either a_i^p or b_i^p is part of an empty switch $\langle -, - \rangle$, then the other value is directly chosen (if both a_i^p and b_i^p are empty switches, the result is also an empty switch). As in the n-point recombination, offspring schedules may be invalid with respect to representation constraints. We use the same procedure as described above after recombination to satisfy these representation constraints.

The mutation operators used for this representation are:

- *Uniform mutation.* The uniform mutation applied to level-controlled triggers is adapted to the absolute time-triggers representation as follows. The domain of a time-controlled trigger c_i^p is restricted to $[c_{i-1}^p + 1, c_{i+1}^p - 1]$, with special cases of

$[0, c_2^p - 1]$ for $i = 1$, and $[c_{2 \cdot SW-1}^p + 1, 24]$ for $i = 2 \cdot SW$. If c_i^p is part of an empty switch, then we replace the empty switch with a new, randomly generated, pair of trigger values.

- *Replace mutation.* This is a more aggressive mutation than *uniform* mutation. It replaces one trigger value with a uniform random integer in the range $[0, T]$. In the special case that the replaced value was part of an empty switch, the whole empty switch (two trigger values) is replaced by generating an additional random integer.

After mutation, representation constraints are enforced by repairing solutions using the same method as for recombination operators. Mutation is applied with a probability such that the expected number of time-triggers modified per solution is two, that is two divided by the number of decision variables. Since the number of decision variables is $2 \cdot SW$ trigger values per pump, we use a mutation probability of $2/(2 \cdot SW \cdot N^{\mathrm{P}})$. This value is larger than the typical one mutation per solution because, after repairing a mutated solution to satisfy representation constraints, the mutated solution may be equal to the original.

4.4 Relative Time-controlled Triggers

The same recombination and mutation operators used for absolute time-controlled triggers are also used for the variant based on *relative time*. Since the representation constraints are slightly different (see Section 3.3.2), the particular implementation of the operators is different as well. In particular, the solution resulting after recombination or mutation may have a total amount of time greater than the scheduling period T, hence violating representation constraints. To repair such solutions, we iteratively reduce the value of a randomly chosen time interval by one time unit until the total sum is equal to T. The recombination operators used for relative time-controlled triggers are:

- *N-point recombination.* This operator follows basically the same procedure as in the absolute time-controlled triggers representation, except for the repair mechanism used if representation constraints are broken. In the experiments, we focus on *one-point* and *two-point* recombination.
- *Uniform recombination.* Because in this case values do not need to keep an increasing order, we have implemented a simpler alternative than for the *absolute* time-controlled triggers. Here, the schedule of pump p in the offspring solution is obtained by randomly selecting, for each trigger, the value of either parent with equal probability, as shown in Fig. 8.

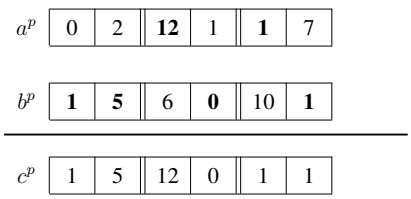

Figure 8: Example of uniform recombination for relative time-controlled triggers representation.

- *Rand-arithmetical recombination.* It is similar to the variant for absolute time-controlled triggers, with the difference that, in this case, we do not handle empty switches in any special way, since they are represented by the value 0.

We use the following mutation operators:

- *Uniform mutation.* For each mutated trigger, another trigger of the same pump is randomly chosen and their total time is redistributed randomly among them. For example, let us assume that mutation is applied to trigger k of a schedule of pump p denoted by c_k^p. First, a different random position j is selected. Next, the value of c_k^p after mutation ($c_k'^p$) is a random integer generated in the interval $[0, c_k^p + c_j^p]$. The new value for trigger j is $c_j'^p = (c_k^p + c_j^p) - c_k'^p$. Figure 9 gives an example with $k = 4$ and $j = 5$.

c^p	1	5	6	**0**	**10**	1
c'^p	1	5	6	6	4	1

Figure 9: Example of uniform mutation for relative time-controlled triggers representation.

- *Replace mutation.* As is the case for other representations, *replace* mutation is meant to be a more aggressive variant of *uniform* mutation. In this case, a trigger value is replaced by an integer randomly generated from the interval $[0, T-(2\cdot SW)]$.

Similar to the case of *absolute time-controlled triggers*, mutation is applied with a probability of $2/(2 \cdot SW \cdot N^{\mathrm{p}})$, that is, two divided by the number of decision variables.

5 Experiments on Different Representations

5.1 Experimental Setup

The simplicity of SEA allows us to focus on the differences between the representations rather than in other algorithmic details. However, it is reasonable to expect performance differences depending on the values of the population size (α), parent/offspring population size (μ), and the particular recombination and mutation operators used. Therefore, we will first study the effect of different parameters for each representation. However, it is not our goal to "*over-tune*" the algorithm for each representation and network instance. Hence, only a few reasonable values of α and μ will be tested in order to identify general trends rather than particular optimal settings.

SEA is tested on two WDNs: the Vanzyl test network and the Richmond network. The former is a network instance designed solely for benchmarking purposes, while Richmond is a medium-sized real-world network instance. The criteria to choose these two networks were the existence of earlier research, hence our results can be compared to previous optimisation algorithms, and the availability of the complete network descriptions, which allows researchers to replicate and extend our experiments. In the original formulation of these network instances, pump operations are triggered at certain water levels of the tanks, which is the common practice in UK. In this paper, we modify the network instances in order to test representations of pump schedules that do not rely on level-controlled triggers.

Vanzyl test network Van Zyl (van Zyl, 2001; van Zyl et al., 2004) proposed this test network as a small yet complex benchmark network to fine-tune the parameters of a hybrid evolutionary algorithm. The layout of the network is shown in Fig. 10. It contains all the main elements of a typical WDN: a source of potable water (reservoir), three pumps, two tanks, and a check valve, which prevents water flowing backwards.

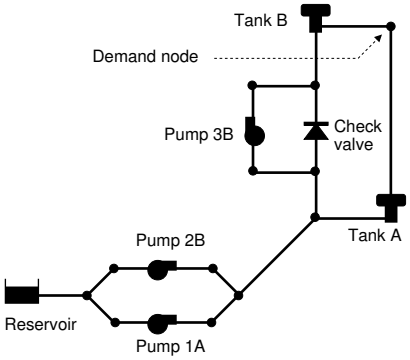

Figure 10: Vanzyl test network.

In this instance the demand charge is taken to be zero and the water available at the reservoir is assumed to be infinite. The electricity cost is divided into two periods with a peak electricity tariff period from 7:00 to 24:00 and a off-peak tariff from 0:00 to 7:00. The demand pattern contains two peaks at 7:00 and 18:00. More details about the test instance are provided by van Zyl et al. (2004) and López-Ibáñez (2009).

Richmond test network Richmond water distribution system is a real system located in the United Kingdom. The network has seven pumps, six tanks and one reservoir. Figure 11 shows a simplified schematic layout that gives an approximate idea of the connections between network elements. The actual network instance used in this work consists of 948 links and 836 nodes. Similarly to the Vanzyl network above, electricity consumption charge is divided into two periods with a peak electricity tariff period from 7:00 to 24:00 and there is no electrical demand charge. A complete description of the Richmond network is available online at `http://iridia.ulb.ac.be/~manuel/ps_instances`. This network was first studied by Atkinson et al. (2000), who applied an evolutionary algorithm to reduce the annual operation cost with respect to the original operational policies based on operator experience. Later, van Zyl et al. (2004) modified the network definition so that all tanks were 95% full at the start of the peak electricity period (7:00 *am*). Recently, López-Ibáñez et al. (2008) have used this latter formulation to assess the applicability of ant colony optimisation to the pump scheduling problem. This is the same network instance used in this work, so results obtained here are directly comparable to those reported by van Zyl et al. (2004) and López-Ibáñez et al. (2008).

In order to obtain a running time comparable to previous works, we use the same termination criteria, that is, 6000 function evaluations per run for the Vanzyl network, and 8000 function evaluations per run for the Richmond network. In the case of a real-world instance like the Richmond network, a single run of 8000 evaluations requires around one hour of CPU-time on a single core of an AMD Opteron(tm) processor 2216HE 2.4 GHz, and the time consumed by hydraulic simulation is always more than 99.9% of the total computation time. Therefore, there is a negligible overhead introduced by the different representations and evolutionary operators.

Since the algorithms are stochastic, in order to assess the average behaviour, each combination of parameters is repeated a number of times with different random seeds. For the Vanzyl network, after some initial testing, 25 repetitions for each parameter combination were considered to provide a sufficiently accurate median value. For the Richmond network the results showed less variability and each run is computationally

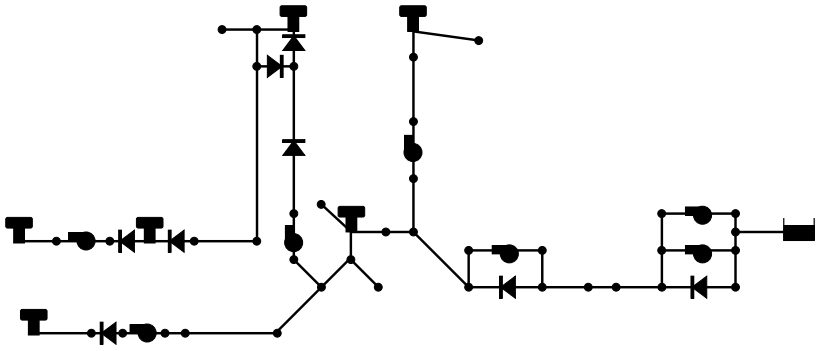

Figure 11: Simplified schematic representation of the Richmond network.

more expensive (the network is much larger and complex). Therefore, 15 runs for each parameter combination were found to be sufficient for assessing the average behaviour.

5.2 Experimental Methodology

The experimental results are analysed by techniques from the field of Experimental Design. Dean and Voss (1999), and Sheskin (2000) cover in detail the techniques used in this paper, but these are standard techniques that may be found in various textbooks on the subject. In particular, experiments are analysed by means of Analysis of Variance (ANOVA). ANOVA identifies which factors (or combination thereof) produce a statistically significant effect on the response variable. In our case, the factors are the parameters of SEA and the response variable is the electrical cost. A factor is significant if the probability of the factor not having an effect on the response variable is less than a given significance level. The p-value of ANOVA (or any other statistical test) is the smallest significance level that would identify the factor as significant. Therefore, low p-values are preferred. It is standard practice to consider a factor (or combination thereof) significant if the corresponding p-value is lower than 0.05 (5%).

ANOVA also indicates whether the effect of one parameter is conditioned by the settings of another parameter. For example, a large population size might compensate for lack of mutation. This is called an interaction and the model analysed includes all possible pairwise interactions among parameters. Differences in mean electrical cost between parameter settings are measured by means of Tukey 's Honest Significant Difference (HSD) 95% confidence intervals (Dean and Voss, 1999). These confidence intervals are incorporated to the *interaction plots* as error bars around the mean electrical cost. We use Fig. 16c as an example. The y-axis gives the electrical cost and the x-axis gives the levels of one factor, in this case the different mutation operators. Points joined by a line are obtained using the same level of a second factor, in this case the recombination operator. The lines in an interaction plot only help to identify points generated with the same level of the second factor, and they do not actually represent any intermediate values or interpolation. Each point represents the mean electrical cost of all runs using a particular combination of mutation and recombination. The error bars are the 95% confidence intervals around each mean. Since the error bars of different recombinations overlap in the case of *uniform* mutation, we conclude that the recombination operators do not have a (statistically significant) different effect on the electrical cost. On the other hand, there is no overlap among the error bars of the recombination operators when using *replace* mutation, and hence, we conclude that, in this case, *one-point* recombination

is significantly better than the other recombination operators.

The correctness of ANOVA depends on several assumptions about the data and the model being analysed (Dean and Voss, 1999). Precise details about their nature and the procedures for checking them are beyond the scope of this paper. Suffice to say that we always check these assumptions before applying ANOVA (López-Ibáñez, 2009). If any of them is not met, we apply standard procedures for correcting the problem. The correction procedures employed in this paper involve removing one or more parameter settings from the experiment (e.g., parameters that result in a strong effect on the electrical cost or large variability). Alternatively, standard transformations of the data may also be employed.

In some cases no transformation or correction would allow us to meet ANOVA's assumptions. In such cases, instead of ANOVA, we use boxplots to compare the parameter settings, and the (possible) interaction between two parameters. Typical boxplots are used to summarise a sample of data values. The line in the middle of the box corresponds to the median value. The "box" is delimited by the first and third quartiles, where the first quartile delimits the lowest 25 percent of the data and the third quartile delimits the lowest 75 percent of the data. Hence, the box contains at least 50 percent of the data. The height of the box corresponds to the inter-quartile range (IQR), which measures the variability of the sample. The extra lines above and below the box are called "whiskers" and they extend to the smallest value (respectively largest value) that is no more than $1.5 \cdot \mathrm{IQR}$ times lower than the first quartile (respectively, higher than the third quartile). Any value beyond the two whiskers is called an outlier and is displayed as a point. Boxplots do not provide confidence levels (it cannot be said that one parameter setting is better than another with a confidence of 95%), however, they are useful to examine the distribution of the data and identify trends. In general, tall boxes (and whiskers) indicate high variability of the results, and the larger overlap between two boxes, the smaller is the difference between two parameter settings. In boxplots of interactions (see Fig. 18 for an example), points correspond to the median electrical cost and only the "box", which contains 50% of the values, of each boxplot is shown. Whiskers and outliers are omitted for clarity. The "boxes" have different widths to appreciate overlapping boxes.

5.3 Binary Representation

The experimental setup of SEA using the binary representation considers all possible combinations (full factorial design (Dean and Voss, 1999)) of $\alpha = \{50, 100, 200\}$, $\mu = \{5, 20\}$, recombination and mutation operators. Recombination can be either *one-point*, *two-point*, or *uniform*, while both *flip* mutation and no mutation are tested.

The constraint on the number of pump switches ($N_p^{\mathrm{sw}} \leq 3$, Eq. 7 with $SW = 3$) is explicitly incorporated to our constraint handling method, as described in Section 2.2.

The Vanzyl Network ANOVA indicates that strong interactions (p-values less than 0.001) exist between mutation and recombination, and between mutation and population size (α), whereas the number of offspring solutions (μ) does not have a significant effect on the electrical cost. These two relevant interactions are shown in Fig. 12, which shows that the best setting for α or recombination operator strongly depends on whether *flip* mutation is used. When using *flip* mutation, the best results are obtained with $\alpha = 50$ and *one-point* recombination, whereas the lowest electrical cost without mutation is obtained with $\alpha = 200$ and *uniform* recombination. If we had started our analysis without mutation, and proceeded to identify the good settings for α and recombination, we would have obtained the wrong answer. The plots also show that using *flip* mutation

is better than no mutation for all combinations of other parameters. In fact, the use of mutation was found to be essential in the four representations tested. Moreover, a few runs without mutation were unable to generate a pump schedule satisfying volume and pressure constraints. Therefore, in the rest of the paper we focus on the results obtained using mutation, taking into account that our complete experimental setup included the lack of mutation, and we tested that it does not lead to good results for any combination of the other parameters (López-Ibáñez, 2009).

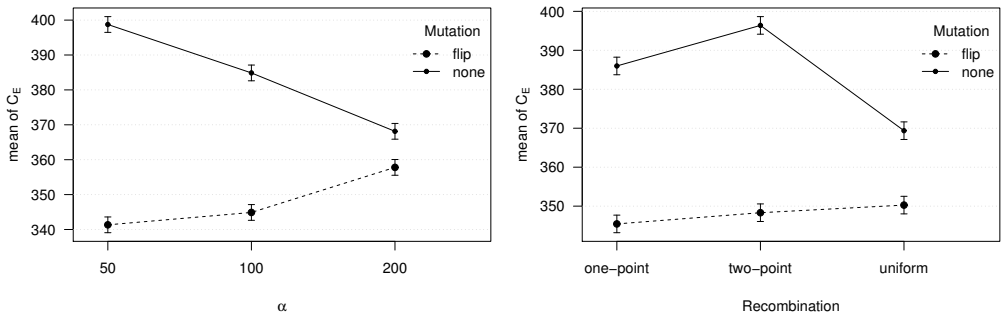

Figure 12: Interaction plots of SEA using the binary representation (the Vanzyl network). (a) Interaction between population size (α) and mutation operator; (b) interaction between recombination operator and mutation operator. The y-axis gives the electricity cost (C_{E}).

The Richmond Network In this case, there is only one statistically significant interaction according to ANOVA between the population size (α) and the recombination operator. This interaction is shown in Fig. 13. The combination of $\alpha = 200$ and *uniform* recombination performs particularly worse than other combinations of parameters. Similar behaviour was also observed in the case of the Vanzyl network. The interaction plot shows that the best value for α is 50. Given $\alpha = 50$, the performance of various recombination operators is not statistically different (the Tukey HSD's intervals overlap).

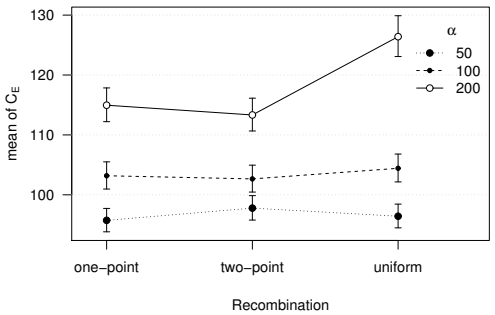

Figure 13: Interaction plot of SEA using the binary representation (the Richmond network). The y-axis gives the electricity cost (C_{E}). The x-axis gives different recombinations. Points joined with a line denote runs using the same population size (α).

5.4 Level-controlled Triggers

For the representation based on level-controlled triggers we perform experiments using *average*, *rand-arithmetical* and *extended-intermediate* recombination operators, and *uniform*, *gaussian*, and *replace* mutation operators. Population sizes of $\alpha = \{50, 100, 200\}$, and offspring population sizes of $\mu = \{5, 20\}$ were used in these experiments. An explicit constraint on the number of pump switches is added so that solutions with more than three switches per pump are penalised ($N_p^{\text{sw}} \leq 3$). This constraint is implemented in the same way as for the binary representation above.

A first observation is that *average* or *rand-arithmetical* recombination generated solutions with volume deficit for at least one run (in many cases, for most of the runs). On the other hand, runs using *extended-intermediate* recombination were always able to obtain solutions with zero volume deficit. Therefore, we restrict our analysis to the results obtained by using *extended-intermediate* recombination. We analyse separately the results for the Vanzyl and Richmond networks to identify which parameters produce a significant effect on the output of SEA.

The Vanzyl Network The only significant factor identified by ANOVA is the mutation operator. In fact, there is statistically significant (but small) advantage of using *replace* mutation over the other two alternatives, *gaussian* and *uniform* mutations, with Tukey HSD 95% confidence intervals for the difference of mean electrical cost values of $[2.6, 6.2]$ and $[1.6, 5.1]$, respectively.

The Richmond Network In the case of the Richmond network, even before attempting ANOVA, the results suggest that the effect of mutation is even stronger. *Gaussian* mutation obtains much higher electrical cost than *replace* and *uniform* mutation, and these differences would actually hide the effect of other parameters. Therefore, we focus on the results obtained using *uniform* and *replace* mutation. We do not apply ANOVA here because the data does not strictly conform to the normality assumption. Even studying the distribution of electrical cost obtained by each combination of parameters (Fig. 14), it is difficult to find a discernible pattern among the parameters. Still, some combinations are better than others, e.g., $\mu = 20$, $\alpha = 50$, and *replace* mutation, obtains better results than using $\alpha = 200$ in most runs. However, the same combination is probably statistically similar to a few others. As before, μ does not appear to have an effect on the results.

5.5 Absolute Time-controlled Triggers

The experimental setup in the case of absolute time-controlled triggers representation is $\alpha = \{50, 100, 200\}$, $\mu = \{5, 20\}$, *uniform*, *one-point*, *two-point* and *rand-arithmetical* recombinations, and *uniform* and *replace* mutations (the full results also consider no mutation).

As was done for the previous representations, the number of switches per pump is constrained to be less than or equal to three ($N_p^{\text{sw}} \leq 3$). However, this constraint does not need to be handled explicitly because the time-controlled triggers representation implicitly enforces this constraint.

The Vanzyl Network ANOVA of the SEA results obtained for the Vanzyl network identifies three interactions that have a significant effect on the electricity cost of the resulting schedule. In particular, the interactions between: recombination operator and population size (α); mutation operator and α; and recombination operator and mutation operator. We explain the overall conclusions obtained from the corresponding interaction plots:

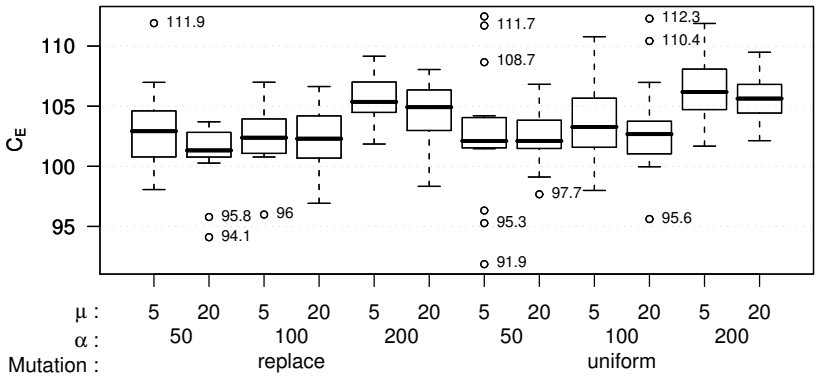

Figure 14: Boxplot of SEA results using level-controlled triggers (the Richmond network). The y-axis gives the electricity cost (C_{E}). The x-axis denotes all possible combinations of parameters mutation, population size (α) and parent/offspring population size (μ) used in the runs summarised by each boxplot.

- **Fig. 15a: Recombination operator and population size (α).** The most evident observation is that *uniform* recombination generates worse results than other recombination operators, independently of the value of α. In the case of *rand-arithmetical* and *two-point* recombination, $\alpha = 200$ is statistically worse than the other possible settings of α. On the other hand, the results of $\alpha = \{50, 100\}$ and *one-point*, *two-point*, and *rand-arithmetical* are not statistically different.

- **Figure 15b: Mutation operator and population size (α).** This plot corroborates the previous observation that $\alpha = 200$ performs significantly worse than the other settings of α. Moreover, the combination of $\alpha = 50$ and *replace* mutation obtains significantly better results than other combinations.

- **Mutation operator and recombination operator.** For brevity we do not show this plot here, It corroborates the above conclusion that *uniform* recombination produces worse results. In addition, it shows a slightly advantage of *one-point* and *two-point* recombination versus *rand-arithmetical*. It does not show any significant difference between the mutation operators.

From the above analysis, the best parameters would be $\alpha = 50$, *replace* mutation, and *one-point* or *two-point* recombination. The offspring population size (μ) does not play a significant role in the performance of SEA when using absolute time-controlled triggers. For further comparison, we choose the configuration with $\alpha = 50$, $\mu = 20$, *two-point* recombination and *replace* mutation. This combination obtains both a low median electrical cost (although not the lowest) and a low variability.

The Richmond Network In the case of the Richmond network, the results of SEA do not satisfy the requirements of ANOVA, even after removing the results corresponding to not using mutation and applying various transformations. We observed that *uniform* recombination produced noticeably worse results than the rest of recombination operators. Excluding those runs using *uniform* recombination from our analysis and checking the ANOVA assumptions again, the data sufficiently meets the requirements and we proceed with ANOVA.

ANOVA identifies the interactions between α and recombination, mutation and α, and mutation and recombination, as having a significant effect on the results. We examine these interactions in more detail in the following:

- **Fig. 16a: Population size (α) and recombination operator.** A small population size ($\alpha = 50$) seems preferable compared to larger values ($\alpha = 200$). Moreover, *one-point* and *two-point* recombination are superior to *rand-arithmetical* recombination when $\alpha = 50$.
- **Figure 16b: Population size (α) and mutation operator.** A small population size generates better results than a setting of $\alpha = 200$. Although there are significant differences between the two mutation operators for large values of α (in favour of *uniform* mutation), this is not true any more for $\alpha = 50$.
- **Figure 16c: Mutation and recombination operators.** *One-point* recombination obtains good results independently of the mutation operator. However, the other recombination operators perform significantly worse when using *replace* mutation than when using *uniform* mutation.

According to the above analysis, the best combination of parameters for SEA when using absolute time-controlled triggers is $\alpha = 50$, *one-point* or *two-point* recombination, and *uniform* or *replace* mutation. Among these possible configurations of SEA, there is no clear winner. For further comparison, we choose the settings $\mu = 5$, *one-point* recombination and *replace* mutation, given its low median value and variability.

5.6 Relative Time-controlled Triggers

The same experiments performed using absolute time-controlled triggers are also performed for relative time-controlled triggers. That is, $\alpha = \{50, 100, 200\}$; $\mu = \{5, 20\}$; *uniform*, *one-point*, *two-point* and *rand-arithmetical* recombination; and *uniform* and *replace* mutation. The number of switches per pump is limited to three ($N_p^{\text{sw}} \leq 3$). This constraint is implicitly handled by the time-controlled triggers representation.

The Vanzyl Network For the Vanzyl network, before applying ANOVA, we notice that the results of SEA with *rand-arithmetical* recombination are better than those obtained using other recombination operators, independent of the other parameters of SEA. In order to confirm this observation, we perform a non-parametric statistical multiple-samples test (Kruskal-Wallis one-way analysis of variance by ranks) (Sheskin, 2000) on the hypothesis that the median electrical cost is the same for the four recombination operators. This hypothesis is rejected by the test with a p-value close to zero. Then we perform pairwise nonparametric two-sample tests (Wilcoxon rank-sum test) on the hypothesis that the median electrical cost is the same for two recombination operators. These tests give a p-value less than 0.05 only when comparing with *rand-arithmetical* recombination. We conclude that the median electrical cost obtained by SEA using *rand-arithmetical* recombination is lower than the median obtained with any other recombination operator. Therefore, we focus on the results obtained using *rand-arithmetical* recombination.

We cannot apply ANOVA because the requirements are not satisfied, even after trying several transformations of the data. In fact, when plotting the distribution of the results (Fig. 17), the only discernible pattern is that $\alpha = 200$ typically produces worse results than using 50 or 100. Apart from those, the apparent differences are very small or are restricted to a particular configuration. For example, either $\alpha = 50$, $\mu = 20$

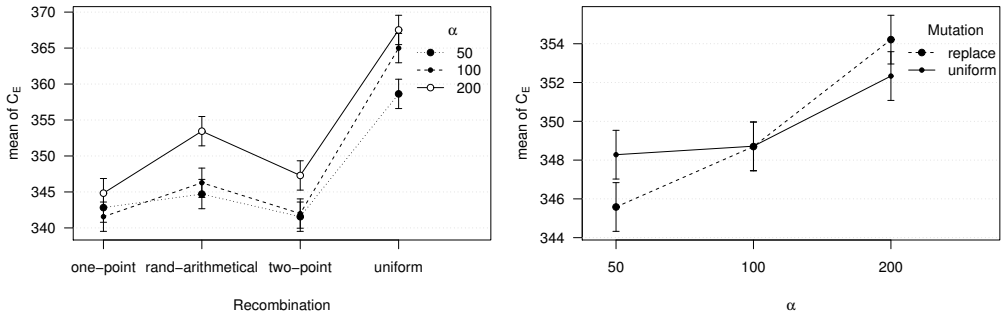

Figure 15: Interaction plots of SEA using absolute time-controlled triggers (the Vanzyl network). (a) Interaction between recombination operators and population size (α); (b) interaction between population size (α) and mutation operator. The y-axis gives the electricity cost (C_{E}).

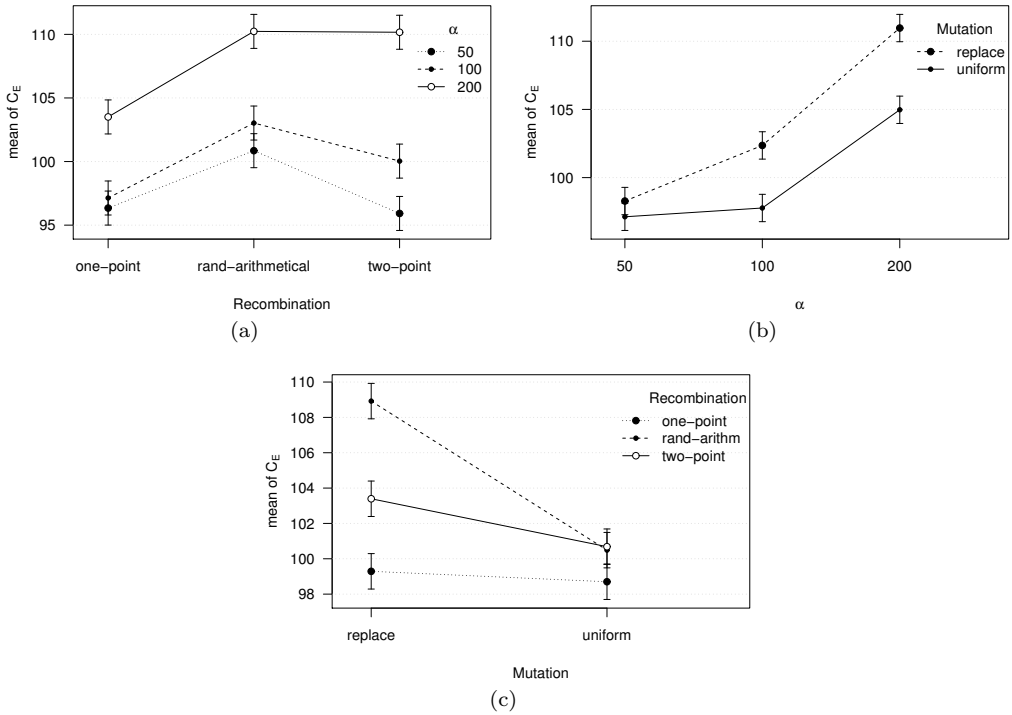

Figure 16: Interaction plots of SEA using absolute time-controlled triggers (the Richmond network): (a) Interaction between recombination operator and population size (α); (b) interaction between population size (α) and mutation operator; (c) interaction between mutation and recombination operators. The y-axis gives the electricity cost (C_{E})

and *replace* mutation, or $\alpha = 100$, $\mu = 20$ and *uniform* mutation have a small median electrical cost. However, we would choose the former because of its small variability, which denotes more consistent results.

The Richmond Network In the Richmond network, in addition to the strong effect caused by *rand-arithmetical* recombination, we observe that also α has a strong influence on the performance of SEA. We examine the combined effect of these two parameters in the boxplot of Fig. 18, where the whiskers and outliers are not plotted for the sake of clarity. This figure shows that there is a substantial degradation of the results as α increases for any recombination operator, although it is less marked for *rand-arithmetical* recombination. Furthermore, given a particular value of α, *rand-arithmetical* is always the best recombination operator.

We therefore focus our analysis on the results obtained using *rand-arithmetical* and $\alpha = \{50, 100\}$. This subset of data does not satisfy ANOVA assumptions. Instead, we examine the results for each configuration of the parameters in Fig. 19. We observe that the results using *replace* mutation and $\alpha = 100$ are worse than other configurations of parameters. On the other hand, the results using *uniform* mutation are only slightly affected by the settings of α and μ. Among these combinations, we choose the combination $\alpha = 50$, $\mu = 5$ and *uniform* mutation for further comparison.

6 Comparison among Representations

In the previous sections we have examined different combinations of the parameters of SEA for each representation and each network instance. In this section, we compare the best combinations of parameters for each representation. We use as a baseline for comparison the results obtained by the Hybrid GA proposed by van Zyl (2001). The Hybrid GA is a steady-state evolutionary algorithm combined with a hill-climber strategy (Hooke & Jeeves method) for optimising level-controlled triggers. The Hybrid GA uses *one-point* crossover and a mutation operator similar to *replace* mutation. Since the representations are different, we are not comparing directly the relative performance of SEA and Hybrid GA, but rather alternative representations using Hybrid GA as a baseline. The results of Hybrid GA shown here are those published by van Zyl et al. (2004), and they were obtained after the same number of evaluations, that is, 6000 evaluations for the Vanzyl network and 8000 for the Richmond network. The Hybrid GA also uses EPANET as the hydraulic simulator, and, hence, results are directly comparable.

The best combinations of parameters of SEA are shown in Table 2 for the Vanzyl network and Table 4 for the Richmond network. Figures 20 and 21 compare graphically the different representations with the results obtained by the Hybrid GA proposed by van Zyl (2001). Finally, Tables 3 and 5 provide the results of pairwise non-parametric Wilcoxon tests (with adjustments for multiple comparisons), which indicate whether the differences observed are statistically significant.

From these results, we conclude that the binary and time-controlled representations outperform both SEA and HybridGA with level-controlled triggers. In addition, SEA with time-controlled triggers using relative time values obtains a lower median electrical cost and a lower variability than using absolute time values. Although the binary representation obtains results similar to relative time-controlled triggers, the latter ensures that the constraint on pump switches is always satisfied. The binary representation has to search for feasible schedules satisfying this constraint. Table 4 indicates that SEA with the binary representation did not find a feasible schedule satisfying $N_p^{\mathrm{sw}} \leq 3$ in at least half of the runs in the Richmond network because the median N^{sw} is larger than

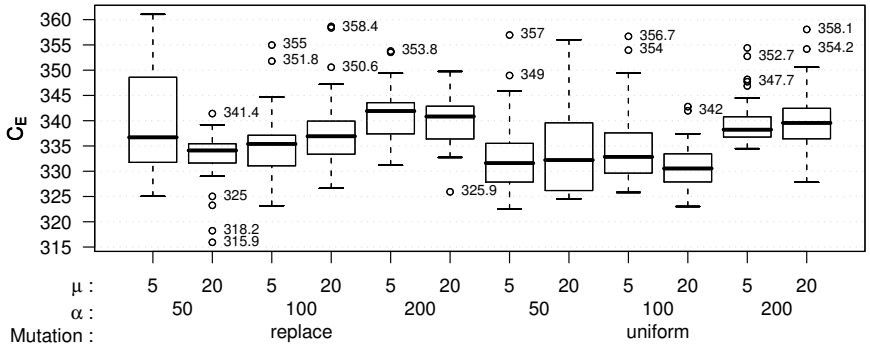

Figure 17: Boxplot of SEA with *rand-arithmetical* recombination using relative time-controlled triggers (the Vanzyl network). The y-axis gives the electricity cost (C_{E}). The x-axis denotes all possible combinations of parameters mutation, population size (α) and parent/offspring population size (μ) used in the runs summarised by each boxplot.

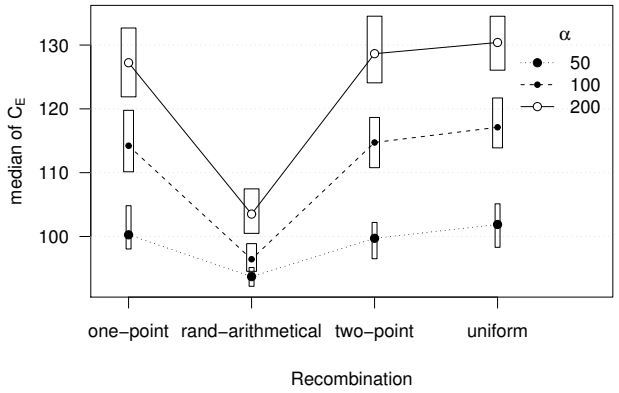

Figure 18: Boxplot of SEA using relative time-controlled triggers (the Richmond network). The y-axis gives the electricity cost (C_{E}). The x-axis denotes different recombinations. Points joined with a line denote runs using the same population size (α).

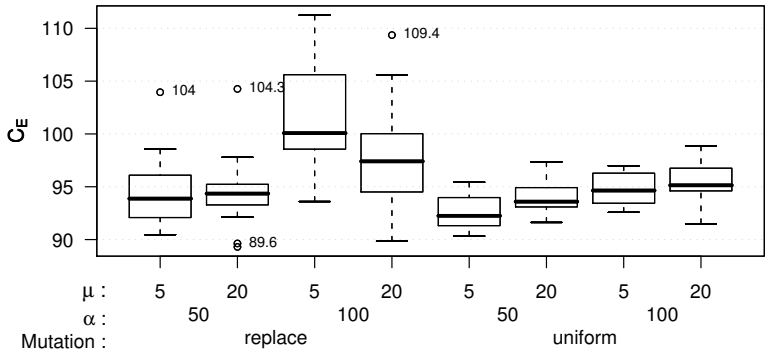

Figure 19: Boxplot of SEA using relative time-controlled triggers with *rand-arithmetical* recombination (the Richmond network). The y-axis gives the electricity cost (C_{E}). The x-axis denotes all possible combinations of parameters mutation, population size (α) and parent/offspring population size (μ) used in the runs summarised by each boxplot.

three times the number of pumps ($N^{\mathrm{p}} = 7$). Both Tables 2 and 4 show that the number of pump switches obtained when using the binary representation is higher than when using relative time-controlled triggers.

Our conclusion is that the proposed relative time-controlled trigger representation is better than level-controlled triggers and the binary representation, with the added benefit that the new time-controlled triggers representations always enforce a maximum number of pump switches. Moreover, the comparison with HybridGA shows that a carefully chosen representation, with the appropriate recombination and mutation operators, may make a simplistic algorithm, such as SEA, outperform a more advanced and complex algorithm, such as HybridGA. These results should not be understood as SEA outperforming Hybrid GA when using the same representation. On the contrary, we believe that adapting Hybrid GA to the time-controlled triggers representation may further improve the results obtained by SEA.

In order to verify the above results, we run the configurations of SEA described in Table 4 on three additional scenarios of the Richmond network:[2]

Scenario #1: Base demands are increased by 10%, modeling a higher water demand.

Scenario #2: Base demands are decreased by 10%, modeling a lower water demand.

Scenario #3: Demand patterns are randomly shuffled, modeling sudden demand changes.

The figures and tables on pages 36, 37 and 38 indicate that the benefits of the time-controlled triggers representations, specially of the relative-time variant, are even greater on these scenarios. In general, the results confirm the conclusions above, that is, binary and time-controlled triggers produce better results than level-controlled triggers, and relative time-controlled triggers produce the best results overall.

7 Conclusion

In this paper, we have formally proposed two new representations based on the concept of time-controlled triggers, and evolutionary operators suited for these new representations. In addition, we have empirically tested these new representations and two traditional representations by means of a simple evolutionary algorithm (SEA). We have found that mutation is an essential ingredient of the success of SEA. We have also identified good recombination operators for each representation, in particular, one-point and two-point recombination for binary representation, extended-intermediate for level-controlled triggers, one-point and two-point for absolute time-controlled triggers, and rand-arithmetical for relative time-controlled triggers. Our analysis also indicated that a small population seems to work better for the network instances tested in this work. However, the trade-off between generations and population size should be more thoroughly investigated in the future. Moreover, SEA uses strong elitism and many generations, as suggested by previous works (Savic et al., 1997; van Zyl et al., 2004), and the use of a much smaller elite size and fewer generations has not been tested.

Finally, we have compared the best configurations of SEA for each representation, and found that the binary and time-controlled triggers representations clearly outperformed the level-controlled triggers representation. In fact, SEA with either of these representations is able to outperform a recently proposed Hybrid GA designed for level-controlled triggers. The expected electrical cost obtained by the new representations

[2]These instances are available at `http://iridia.ulb.ac.be/~manuel/ps_instances`

based on time-controlled triggers were not found to be statistically different from the electrical cost obtained by the binary representation. However, the new representations obtained schedules with a much lower number of pump switches, which in turn should reduce maintenance costs.

We conclude that, where physically and practically possible, the use of time-controlled triggers should be considered an alternative to traditional level-controlled triggers. Nonetheless, more work is needed to confirm that the presented results may be generalised to other network instances and optimisation algorithms. The time-controlled triggers representation is not tied to the simple EA tested here. We believe that state-of-the-art EAs will likely enhance our results. The discrete nature of the time-controlled triggers representation means that this representation may be used with metaheuristics such as ant colony optimisation and stochastic local search. Other optimisation problems, such as the unit commitment problem in electricity generation systems (Michalewicz et al., 1996; Ting et al., 2003), share many similarities with pump scheduling, and, hence, the work presented here may be adapted to those problems.

We hope that the detailed methodology and analysis described here will encourage engineers to take into account the interactions among algorithmic parameters when studying the application of an algorithm. We have empirically shown that these interactions exist in real-world instances and may lead to wrong conclusions if not taken into account. We also believe that comparative studies are more scientific and meaningful than single-instance and single-algorithm application studies, because comparative studies allow to identify good algorithms or algorithmic settings for the pump scheduling problem in general, and, therefore, provide engineers facing a new network instance with valuable information.

References

Atkinson, R., van Zyl, J. E., Walters, G. A., and Savic, D. A. (2000). Genetic algorithm optimisation of level-controlled pumping station operation. In *Water network modelling for optimal design and management*, pages 79–90. Centre for Water Systems, Exeter, U.K.

Chase, D. V. and Ormsbee, L. E. (1993). Computer-generated pumping schedules for satisfying operation objectives. *J. Am. Water Works Assoc.*, 85(7):54–61.

Cohen, G. (1982). Optimal control of water supply networks. In Tzafestas, S. G., editor, *Optimization and Control of Dynamic Operational Research Models*, volume 4, chapter 8, pages 251–276. North-Holland Publishing Company, Amsterdam.

Dandy, G. C. and Gibbs, M. S. (2003). Optimizing system operations and water quality. In Bizier, P. and DeBarry, P., editors, *Proceedings of World Water and Environmental Resources Congress*. ASCE, Philadelphia, USA. on CD-ROM.

Dean, A. and Voss, D. (1999). *Design and Analysis of Experiments*. Springer, London, UK.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2/4):311–338.

Eshelman, L. J. and Schaffer, J. D. (1992). Real-coded genetic algorithms and interval-schemata. In Whitley, L. D., editor, *FOGA*, pages 187–202. Morgan Kaufmann Publishers.

Farmani, R., Walters, G. A., and Savic, D. A. (2006). Evolutionary multi-objective optimization of the design and operation of water distribution network: total cost vs. reliability vs. water quality. *Journal of Hydroinformatics*, 8(3):165–179.

Fogel, D. B. (1995). *Evolutionary Computation. Toward a New Philosophy of Machine Intelligence*. IEEE Press.

Goldman, F. E. and Mays, L. W. (2000). The application of simulated annealing to the optimal operation of water systems. In *Proceedings of 26th Annual Water Resources Planning and Management Conference*, Tempe, USA. ASCE.

Herrera, F., Lozano, M., and Sánchez, A. M. (2003). A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study. *International Journal of Intelligent Systems*, 18(3):309–338.

Jowitt, P. W. and Germanopoulos, G. (1992). Optimal pump scheduling in water supply networks. *Journal of Water Resources Planning and Management, ASCE*, 118(4):406–422.

Kazantzis, M. D., Simpson, A. R., Kwong, D., and Tan, S. M. (2002). A new methodology for optimizing the daily operations of a pumping plant. In *Proceedings of 2002 Conference on Water Resources Planning*, Roanoke, USA. ASCE.

Lansey, K. E. and Awumah, K. (1994). Optimal pump operations considering pump switches. *Journal of Water Resources Planning and Management, ASCE*, 120(1):17–35.

Leon, C., Martin, S., Elena, J. M., and Luque, J. (2000). EXPLORE: Hybrid expert system for water networks management. *Journal of Water Resources Planning and Management, ASCE*, 126(2):65–74.

López-Ibáñez, M. (2009). *Operational Optimisation of Water Distribution Networks*. PhD thesis, School of Engineering and the Built Environment, Edinburgh Napier University, UK.

López-Ibáñez, M., Prasad, T. D., and Paechter, B. (2008). Ant colony optimisation for the optimal control of pumps in water distribution networks. *Journal of Water Resources Planning and Management, ASCE*, 134(4):337–346.

Mäckle, G., Savic, D. A., and Walters, G. A. (1995). Application of genetic algorithms to pump scheduling for water supply. In *Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA'95*, volume 414, pages 400–405, Sheffield, UK. IEE Conference Publication.

Maier, H. R., Simpson, A. R., Zecchin, A. C., Foong, W. K., Phang, K. Y., Seah, H. Y., and Tan, C. L. (2003). Ant colony optimization for design of water distribution systems. *Journal of Water Resources Planning and Management, ASCE*, 129(3):200–209.

McCormick, G. and Powell, R. S. (2003a). Optimal pump scheduling in water supply systems with maximum demand charges. *Journal of Water Resources Planning and Management, ASCE*, 129(5):372–379.

McCormick, G. and Powell, R. S. (2003b). A progressive mixed integer-programming method for pump scheduling. In Maksimović, C., Butler, D., and Memon, F. A., editors, *Advances in Water Supply Management*, pages 307–313, London, UK.

McCormick, G. and Powell, R. S. (2004). Derivation of near-optimal pump schedules for water distribution by simulated annealing. *Journal of the Operational Research Society*, 55(7):728–736.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs, 3rd Edition*. Springer, Berlin, Germany.

Michalewicz, Z., Dasgupta, D., Riche, R. G. L., and Schoenauer, M. (1996). Evolutionary algorithms for constrained engineering problems. *Computers and Industrial Engineering*, 30(4):851–870.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). Predictive models for the breeder genetic algorithm. *Evolutionary Computation*, 1(1):25–49.

Nitivattananon, V., Sadowski, E. C., and Quimpo, R. G. (1996). Optimization of water supply system operation. *Journal of Water Resources Planning and Management, ASCE*, 122(5):374–384.

Ormsbee, L. E. and Lansey, K. E. (1994). Optimal control of water supply pumping systems. *Journal of Water Resources Planning and Management, ASCE*, 120(2):237–252.

Ormsbee, L. E. and Reddy, S. L. (1995). Nonlinear heuristic for pump operations. *Journal of Water Resources Planning and Management, ASCE*, 121(4):302–309.

Pezeshk, S. and Helweg, O. J. (1996). Adaptative search optimisation in reducing pump operation costs. *Journal of Water Resources Planning and Management, ASCE*, 122(1):57–63.

Prasad, T. D. (2009). Design of pumped water distribution networks with storage. *Journal of Water Resources Planning and Management, ASCE*, 136(4):129–136.

Rao, Z. and Salomons, E. (2007). Development of a real-time, near-optimal control process for water-distribution networks. *Journal of Hydroinformatics*, 9(1):25–37.

Rossman, L. A. (1999). The EPANET Programmer's Toolkit for analysis of water distribution systems. In *Proceedings of the Annual Water Resources Planning and Management Conference*, Reston, USA. ASCE.

Rossman, L. A. (2000). *EPANET 2 Users Manual*. U.S. Environmental Protection Agency, Cincinnati, USA.

Sakarya, A. B. A. and Mays, L. W. (2000). Optimal operation of water distribution pumps considering water quality. *Journal of Water Resources Planning and Management, ASCE*, 126(4):210–220.

Savic, D. A., Walters, G. A., and Schwab, M. (1997). Multiobjective genetic algorithms for pump scheduling in water supply. In Corne, D. and Shapiro, J. L., editors, *Evolutionary Computing Workshop, AISB'97*, volume 1305 of *Lecture Notes in Computer Science*, pages 227–236. Springer-Verlag Berlin.

Sheskin, D. J. (2000). *Hanbook of Parametric and Nonparametric Statistical Procedures*. Chapman & Hall/CRC, second edition.

Sotelo, A., von Lücken, C., and Barán, B. (2002). Multiobjective evolutionary algorithms in pump scheduling optimisation. In Topping, B. H. V. and Bittnar, Z., editors, *Proceedings of the Third International Conference on Engineering Computational Technology*. Civil-Comp Press, Stirling, Scotland.

Ting, T.-O., Rao, M., Loo, C., and Ngu, S. (2003). Solving unit commitment problem using hybrid particle swarm optimization. *Journal of Heuristics*, 9(6):507–520.

van Zyl, J. E. (2001). *A Methodology for Improved Operational Optimization of Water Distribution Systems*. PhD thesis, School of Engineering and Computer Science, University of Exeter, UK.

van Zyl, J. E., Savic, D. A., and Walters, G. A. (2004). Operational optimization of water distribution systems using a hybrid genetic algorithm. *Journal of Water Resources Planning and Management, ASCE*, 130(2):160–170.

Vasan, A. and Simonovic, S. P. (2010). Optimization of water distribution network design using differential evolution. *Journal of Water Resources Planning and Management, ASCE*, 136(2):279–287.

Walski, T. M., Chase, D. V., Savic, D. A., Grayman, W., Beckwith, S., and Koelle, E. (2003). *Advanced Water Distribution Modeling and Management*. Haestad Methods, Inc., Haestad Press, first edition.

Wegley, C., Eusuff, M., and Lansey, K. E. (2000). Determining pump operations using particle swarm optimization. In Hotchkiss, R. H. and Glade, M., editors, *Building Partnerships, Proceedings of the Joint Conference on Water Resources Engineering and Water Resources Planning and Management*, Minneapolis, USA.

Yu, G., Powell, R. S., and Sterling, M. J. H. (1994). Optimized pump scheduling in water distribution systems. *J. Optim. Theory Appl.*, 83(3):463–488.

Table 2: Comparison of SEA using different representations in the Vanzyl network.

		Representation			
		binary	**level-triggers**	**time-triggers**	
				absolute	**relative**
SEA	α	50	50	50	50
	μ	5	5	20	20
	Recomb.	one-point	ext.-interm.	two-point	rand-arithm.
	Mutation	flip	replace	replace	replace
C_{E}	median	333.0	346.9	338.7	334.1
	sd	11.2	5.3	5.6	6.1
	min	324.7	337.2	325.3	315.9
	max	359.6	357.1	351.2	341.4
N^{sw}	median	7.0	3.0	6.0	5.0
	sd	1.2	0.9	1.3	1.2
	min	5.0	2.0	4.0	3.0
	max	9.0	6.0	8.0	7.0

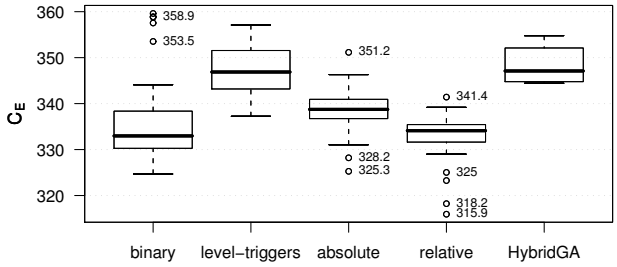

Figure 20: Boxplot of the results of SEA for different representations and Hybrid GA in the Vanzyl network.

Table 3: P-values of Wilcoxon rank sum tests with Holm adjustment for pairwise comparisons. If the difference between the representations is statistically significant at a significance lower than 0.01, it is marked in bold-face.

	binary	level-triggers	absolute	relative
level-triggers	**0.00240**	-	-	-
absolute	0.20367	**7.0e-06**	-	-
relative	0.74370	**7.1e-12**	**0.00240**	-
HybridGA	0.05260	0.64726	**0.00027**	**5.3e-06**

Table 4: Comparison of SEA using different representations for the Richmond network.

		Representation			
		binary	**level-triggers**	**time-triggers**	
				absolute	**relative**
SEA	α	50	50	50	50
	μ	5	20	5	5
	Recomb.	one-point	ext.-interm.	one-point	rand-arithm.
	Mutation	flip	replace	replace	uniform
C_{E}	median	93.8	100.0	95.5	92.3
	sd	2.6	2.5	3.4	1.6
	min	91.4	99.1	90.4	90.3
	max	100.2	107.0	104.2	95.4
N^{sw}	median	26.0	10.0	13.0	16.0
	sd	3.1	1.5	1.1	1.7
	min	19.0	8.0	12.0	14.0
	max	32.0	13.0	15.0	20.0

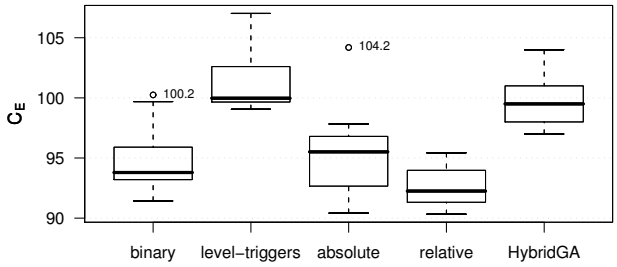

Figure 21: Boxplot of the results of SEA for the Richmond network for different representations.

Table 5: P-values of Wilcoxon rank sum tests with Holm adjustment for pairwise comparisons. If the difference between the representations is statistically significant at a significance lower than 0.01, it is marked in bold-face.

	binary	level-triggers	absolute	relative
level-triggers	**3.8e-05**	-	-	-
absolute	0.71297	**3.8e-05**	-	-
relative	0.07421	**1.3e-07**	0.07421	-
HybridGA	**0.00065**	0.38148	**0.00245**	**5.5e-06**

Table 6: Comparison of SEA using different representations for the Richmond network (Scenario #1).

		Representation			
		binary	**level-triggers**	**time-triggers**	
				absolute	**relative**
C_{E}	median	111.6	119.7	112.8	108.0
	sd	2.4	2.0	2.3	1.8
	min	108.4	113.5	108.9	105.3
	max	116.7	121.7	117.4	112.0
N^{sw}	median	25.0	13.0	15.0	15.0
	sd	2.9	1.6	1.4	1.9
	min	22.0	9.0	12.0	12.0
	max	33.0	15.0	17.0	19.0

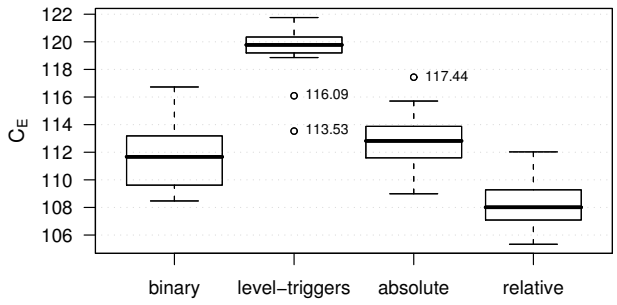

Figure 22: Boxplot of the results of SEA for the Richmond network (Scenario #1) for different representations.

Table 7: P-values of Wilcoxon rank sum tests with Holm adjustment for pairwise comparisons. If the difference between the representations is statistically significant at a significance lower than 0.01, it is marked in bold-face.

	binary	level-triggers	absolute
level-triggers	**7.7e-07**	-	-
absolute	0.28544	**2.3e-06**	-
relative	**0.00018**	**7.7e-08**	**3.5e-05**

Table 8: Comparison of SEA using different representations for the Richmond network (Scenario #2).

		Representation			
		binary	**level-triggers**	**time-triggers**	
				absolute	**relative**
C_{E}	median	80.6	83.1	77.1	75.2
	sd	3.9	1.6	3.5	2.0
	min	76.1	80.1	74.9	71.6
	max	88.0	85.8	86.8	79.1
N^{sw}	median	26.0	11.0	14.0	15.0
	sd	3.6	1.3	1.3	1.4
	min	18.0	9.0	12.0	12.0
	max	33.0	15.0	16.0	18.0

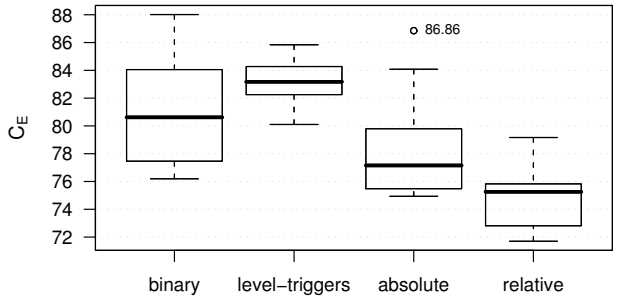

Figure 23: Boxplot of the results of SEA for the Richmond network (Scenario #2) for different representations.

Table 9: P-values of Wilcoxon rank sum tests with Holm adjustment for pairwise comparisons. If the difference between the representations is statistically significant at a significance lower than 0.01, it is marked in bold-face.

	binary	level-triggers	absolute
level-triggers	0.16069	-	-
absolute	0.08168	**0.00086**	-
relative	**6.3e-06**	**7.7e-08**	0.01970

Table 10: Comparison of SEA using different representations for the Richmond network (Scenario #3).

		Representation			
		binary	**level-triggers**	**time-triggers**	
				absolute	**relative**
C_{E}	median	95.5	94.9	94.2	89.8
	sd	2.9	3.0	2.5	2.5
	min	90.0	92.2	90.5	86.7
	max	99.6	104.8	99.2	95.0
N^{sw}	median	26.0	10.0	14.0	15.0
	sd	3.1	1.0	2.1	1.8
	min	19.0	9.0	9.0	13.0
	max	30.0	13.0	18.0	19.0

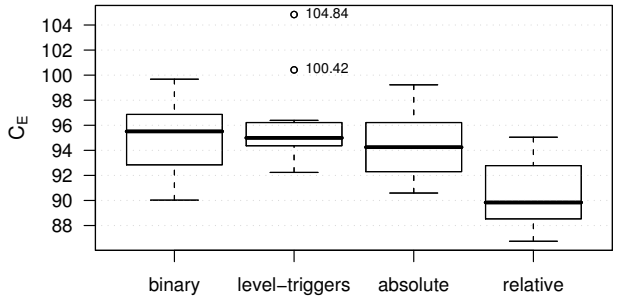

Figure 24: Boxplot of the results of SEA for the Richmond network (Scenario #3) for different representations.

Table 11: P-values of Wilcoxon rank sum tests with Holm adjustment for pairwise comparisons. If the difference between the representations is statistically significant at a significance lower than 0.01, it is marked in bold-face.

	binary	level-triggers	absolute
level-triggers	1.00000	-	-
absolute	1.00000	0.91377	-
relative	**0.00045**	**2.1e-05**	**0.00129**